Sistema para el seguimiento de la posición tridimensional de manos

Marcelo J. Fernández[1], Nicolás Sugino[2], and Alfredo N Campos[3]

Universidad Tecnológica Nacional, Facultad Regional Buenos Aires, Departamento de Electrónica, Medrano 951, Ciudad de Buenos Aires, Argentina,
[1]joako232002@gmail.com,
[2]sugino.nicolas@gmail.com,
[3]acampos@electron.frba.utn.edu.ar

Resumen La interacción entre humanos y computadoras se encuentra en constante evolución hacia la reducción del hardware y la naturalización del uso de los dispositivos. Los métodos de procesamiento de imagen permiten delegar el trabajo a las computadoras aprovechando su enorme capacidad. La mano humana es el objeto de interés de este trabajo donde se propone una solución que permita ubicar los dedos del usuario en tres dimensiones por medio de cámaras web tradicionales, minimizando el hardware y desarrollada en C/C++. El método propuesto combina el uso de envolventes convexas para ubicar los dedos en 2 dimensiones, con la generación de mapas de disparidad por medio de estereoscopía para lograr la tridimensionalidad. Los resultados preliminares demostraron la capacidad de realizar el proceso en tiempo real, con la precisión adecuada y funcionando correctamente aún bajo condiciones desfavorables.

Index terms— Deteccion de Manos, Gestos, Procesamiento de imágenes, Estereoscopía, Envolvente convexa, Mapa de disparidad, Biometría

1. Introducción

Los nuevos dispositivos electrónicos buscan mejorar la interfaz entre el usuario y la computadora o máquina (HCI o HMI: *Human Computer Interaction* o *Human Machine Interaction*) [1]. Por esto cada vez se reduce el uso de teclados y mouse como interfaz primaria ya que estos no proveen la sensibilidad suficiente para ser utilizados como interfaz para dispositivos móviles, sistemas de realidad virtual o aumentada; y los joysticks requieren esfuerzo de adaptación y aprendizaje. En cambio, se opta por métodos de identificación y seguimiento de partes del cuerpo (pantallas táctiles, cascos y guantes de realidad virtual, reconocimiento de voz, entre otros), los cuales resultan no sólo satisfacen las características técnicas, sino que también resultan más "naturales" en su uso.

Los actuales sistemas de HMI pueden ser divididos en dos categorías: los *wereables* o vestibles que utilizan hardware en contacto con el cuerpo humano; y los que utilizan cámaras o láseres. Entre los primeros se puede mencionar el sistema Myo [2] que permite interpretar gestos midiendo la actividad de las articulaciones de las manos o sistemas experimentales, como el desarrollado en base

a guantes para generar mapas auto organizados [3]. Entre los segundos sistemas se puede destacar productos comerciales como el Kinect [4] de Microsoft que combina el uso de un láser y cámara infrarrojos para lograr mapas de disparidad, y el producto Leap Motion [5] que integra 3 emisores LED infrarrojos y dos cámaras para buscar la posición de las manos; y por otro lado trabajos de investigación, como el basado en el reconocimiento de gestos para videojuegos [6] o el reconocimiento de la mano basado en su curvatura para el control de sillas de ruedas [7]. Aunque estas dos categorías de sistemas coinciden en el uso de hardware específico, es posible encontrar marcar que los dispositivos vestibles suelen resultar incómodos [8] y deben ser adaptables a diferentes tipos de personas, mientras que los basados en procesamiento de imágenes generalmente no integran el eje de profundidad en su funcionamiento.

En este trabajo se propuso una solución que minimice el hardware, utilizando solamente una computadora y 2 cámaras; y programada en un lenguaje de bajo nivel logrando así la detección en tiempo real

2. Implementación

El desarrollo se realizó principalmente sobre PC (arquitectura x86), utilizando un procesador Intel i5 6600K. El programa se realizó en C++, además se utilizó la librería OpenCV para acelerar el desarrollo en distintas partes indicadas en la sección Métodos.

Se utilizaron cámaras Ps Eye originalmente diseñadas para la consola de mesa PS3 de la compañía Sony. La resolución de captura fue de 640 x 480. El FOV(*Field Of View*) es ajustable y se utilizó 75 grados. La cámara cuenta con un sensor CMOS (*Complimentary Metal-Oxide Semiconductor*) OV7720 cuya área de imagen es de $3984\mu m$ x $2952\mu m$.

3. Métodos

El método utilizado consistió en la combinación de dos tecnologías: Mapas de disparidad y defectos de convexidad en una envolvente convexa. El proceso comenzó a partir de la captura de 2 imágenes estereoscópicas, a las cuales se les aplicó un filtrado inicial para eliminar ruidos y luego se las rectificó para poder alinearlas. A continuación el proceso se dividió en 2 partes, por un lado utilizando el principio de estereoscopía y las imágenes rectificadas, se obtuvieron mapas de disparidad para saber la distancia de los distintos puntos de las imágenes, este método se detalla en [9]. Por otro lado se trabajó sobre una sola imagen rectificada sobre la que se realizó una segmentación para detectar la mano, se armó una envolvente convexa, y luego se identificaron los dedos entre los defectos de convexidad de la envolvente.

En el reconocimiento de gestos de la mano se consideraron 7 gestos básicos identificados en inglés como *open, point, grasp, fist, v-pose, contain* y *pinch*. El estudio se centró principalmente en el gesto de mano abierta -*open*- y luego se corroboró el funcionamiento con el resto. Finalmente, la ubicación de los dedos

marcan las coordenadas x e y del mapa de disparidad sobre el cual se obtiene la coordenada z y de esta manera, la distancia. En la figura 1 podemos ver el esquema resumido de todo el proceso realizado.

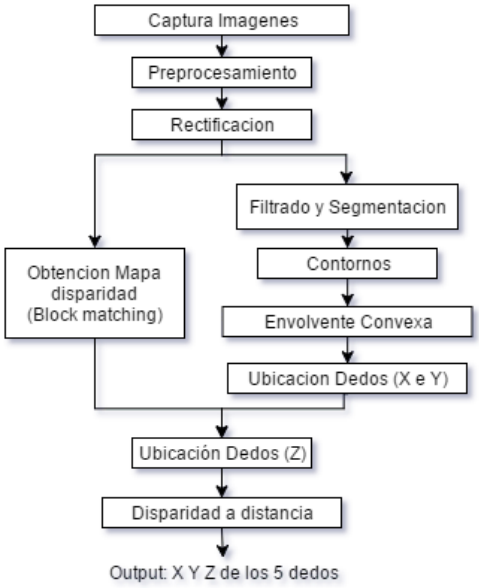

Figura 1. Diagrama en bloques del proceso

3.1. Captura de imagen estereoscópica

Como punto de partida se realizó la captura de 1 cuadro de cada una de las cámaras en formato crudo de 640x480 con una profundidad de 8 bits no signados.

Luego se aplicó a las imágenes obtenidas una convolución con una matriz de 3x3 de tipo gausseana a modo de filtro para reducir el ruido. Se les realizó una rotación y espejado a las imágenes para solucionar el problema de la disposición de cámaras y con esto ya se obtuvieron las imágenes listas para utilizarse tanto en la calibración, en la generación del mapa de disparidad, o detección de dedos.

3.2. Calibración

Para la calibración se utilizó una imagen plana con apariencia similar a un tablero de ajedrez que permite, por medio de algoritmos ya implementados en OpenCV [10], detectar y generar las matrices de corrección de perspectiva para

las cámaras. Este método permite el uso de cámaras económicas que presentan problemas de distorsión en la imagen y también eliminar diferencias que pueda haber entre cámaras de modelo idéntico. Esto es fundamental para luego realizar la detección de disparidad entre imágenes.

En el proceso de calibración se tomaron un numero de fotogramas tratando de ubicar el tablero de ajedrez en distintas posiciones y rotaciones, el proceso se puede reducir a dos etapas *AddSample* y *Calibrate*. Durante *AddSample* se analizó cada imagen de calibración, donde se buscó los extremos y los puntos del tablero. Se requiere tomar imágenes del tablero en distintos ángulos, durante la implementación se utilizaron 5 o 6 capturas. Luego durante el proceso de *Calibrate* se estima la matrices necesarias para la calibración (matrices de cámaras, de distorsión, rotación y traslación).

3.3. Rectificación y generación de mapa de disparidad

Para la generación del mapa de disparidad se utilizó un algoritmo del tipo "Block Matching", estos son frecuentemente utilizados en compresión de video [11]. Este algoritmo se encuentra implementado para generación de mapas de disparidad en OpenCV [12] y como salida se obtuvo un mapa de disparidad, el cual es una imagen de 16 bits de profundidad que representa la disparidad entre las 2 imágenes.

3.4. Conversión de disparidad a distancia

La ecuación 1 nos indica el tipo de relación que hay entre la distancia y la disparidad, donde T es la distancia entre los centros de los sensores CMOS, f es la distancia focal de la cámara donde se esta proyectando, y x^l, x^r son las distancias en pixeles desde el centro de los sensores CMOS hasta donde se proyecta un punto de la imagen en particular. Esto indica que cuando la disparidad

$$Z = \frac{fT}{x^l - x^r} \ . \tag{1}$$

tiende a cero la distancia es infinita y a mayor disparidad mas cercanía al eje del sistema de cámaras. Si bien se dispone exactamente de los valores de f y T, y la disparidad es proporcional a $x^l - x^r$, el procedimiento fue empírico. Se ubicaron objetos a diversas distancias y se verificó la disparidad obtenida, finalmente se armó la curva que pone distancia en función de disparidad, una vez obtenida la misma y si no se modifica el setup el se puede obtener para cada valor de disparidad la distancia correspondiente.

3.5. Filtrado y segmentación

Este proceso comienza a partir de una de las imagenes rectificadas (fig 1). En primer lugar se aplicó un desenfoque gaussiano, el tamaño del kernel que se utilizó es de 3x3, el desenfoque suaviza la imagen para disminuir ruidos que puedan surgir por la cámara o la iluminación. Luego se convirtió la imagen al espacio de color HSV, este espacio de color separa el color (*Hue*) de la luminancia que queda referenciada por 2 valores (*Saturation* y *Value*), logrando que el segmentado se vuelva mas robusto ante cambios de iluminación ya que se acota el matiz [13]. Posteriormente tomando al usuario como ejemplo se determinó un rango de valores para la piel en distintas condiciones de iluminación, el color queda caracterizado por el matiz, se notó el rango mas estrecho para esta variable.

Utilizando este rango de valores se hizo un umbral de la imagen quedándose solamente con pixeles de color similar a la piel humana, posteriormente se aplicó una apertura para unir los *blobs* mas amplios y un filtro de mediana para reducir el ruido tipo *Salt&Pepper* que se puede producir por el umbral (figura 2A).

3.6. Detección de contornos

De la imagen filtrada y segmentada se buscaron todos los contornos externos, la imagen típica tiene una mano en su centro y por ende solo se consideró el contorno que envuelve la mayor área. Por último se aproximó el contorno a un polígono para poder reducir la cantidad de segmentos y vértices.

3.7. Envolvente convexa y ubicación de la palma de la mano

Una vez obtenidos los contornos, para poder comprender la forma de un posible objeto de una manera simple se realizó una envolvente convexa y luego se obtuvieron los defectos de convexidad de la envolvente. Se conoce como envolvente convexa a la región mas pequeña que envuelve a todos los puntos de un conjunto. Por medio de la envolvente convexa y el contorno se pudo ubicar los defectos de convexidad, como se muestra en la figura 2, los defectos de convexidad proveen una manera muy directa de obtener la posición y gesto de la mano. Teniendo los defectos de convexidad se promedió la posición en x e y de los mismos y con esto se obtiene el centro de la palma y promediando la distancia entre los defectos y el centro de la palma se pudo tener una estimación del radio de la palma. Para el reconocimiento de gestos se puede incorporar lo planteado por [14], [15] o [16], que quedó fuera del alcance de este trabajo.

3.8. Ubicación y validación de dedos

Trabajando sobre el contorno, se ubicaron máximos locales de distancia entre el centro de la palma y el contorno, estos máximos son los que fueron considerados como dedos. Además se agregó un filtrado por ángulo de los supuestos dedos para eliminar casos imposibles (solo el meñique y el pulgar pueden llegar a estar separados 180 grados y todo el resto de los dedos debían estar comprendidos

entre ellos), además los casos con una cantidad de dedos reconocidos mayor a 5 no fueron validados.

Combinando el filtrado HSV, la selección del contorno que envuelve la mayor área, la limitación de los defectos de convexidad (hasta 5 como máximo) y la validación de ángulos entre los mismos, se logró hacer una correcta diferenciación entre una mano y cualquier otro objeto (fig 2 C).

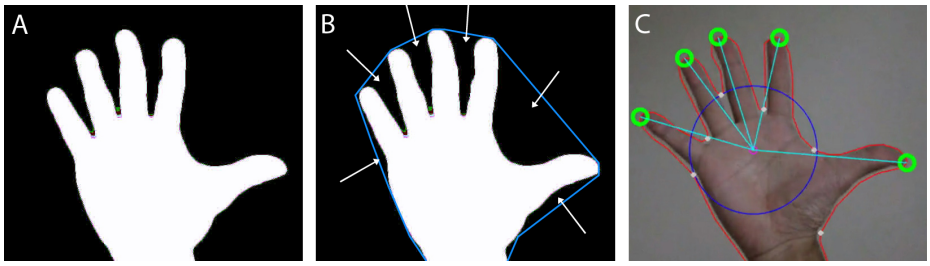

Figura 2. Etapas del proceso de detección de dedos: segmentación (A), envolvente convexa (B), y ubicación de los dedos (C)

4. Resultados y mediciones

Para convertir las unidades de disparidad a longitud en milímetros, se ubicaron objetos a diversas distancias del eje de las cámaras y se armó la curva correspondiente(figura 3). Por último, se obtuvo la curva que permite obtener distancia a partir de la disparidad (ecuación 2).

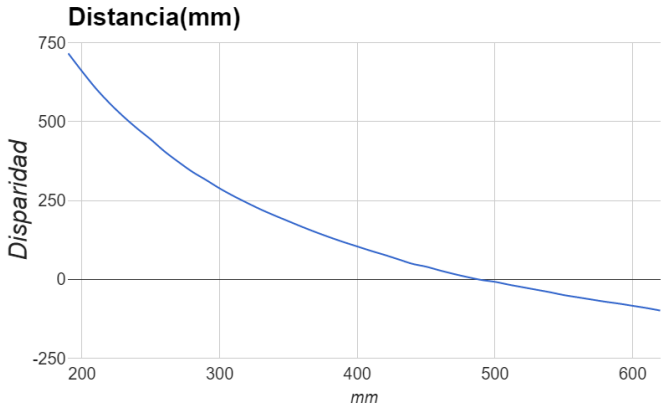

Figura 3. Disparidad en función de distancia

$$Z[mm] = \frac{237125.4}{disp + 476} - 8.925 \ . \tag{2}$$

A distancias próximas (200-300mm) donde la pendiente es mayor, hay mayor resolución, pero esto también se tradujo en mayor variacion de disparidad (mas ruido). En contraparte, a distancias mayores (500-600mm) la resolución es menor, pero el ruido también disminuyó.

Cuadro 1. Mediciones de disparidad y distancias

Distancia[mm]	Disparidad	Distancia[mm]	Incertidumbre[mm]
200	621.2	207	0.039
250	419.8	255	0.039
300	280.1	304	0.15
350	177.6	353	0.091
400	97.6	404	0.15
450	35.1	455	0.091
500	-16.4	507	0.18
550	-58.8	559	0.27
600	-93.6	611	0.26

Se colocó la mano en un punto central de la imagen y se registró a distintas distancias la disparidad medida junto con su desvío, se buscó corroborar la curva levantada anteriormente y ver el error que se comete. Se observó (cuadro 1) un error en la distancia a medida que el objeto se aleja del eje de las cámaras. Por otro lado, la incertidumbre de la medición no muestra un error de tipo estadístico que varíe con la distancia. Si consideramos la calibración hasta los 600mm, el error es inferior al 4 %, siendo de 11mm en el peor caso.

También se midió las indicaciones de disparidad sobre un objeto de frente plano (aproximadamente 20 x 20 cm), en distintos puntos del mismo. Esto se hizo a 2 distancias distintas y se tomaron 20 mediciones, observando los resultados en el cuadro 2 comprobamos como al tener una mayor resolución cerca de la cámara, el ruido aumenta, esto se ve claramente en la incertidumbre de los valores medidos, tener en cuenta que en este caso la incertidumbre esta informada sobre la disparidad ya que se busca una demostración cualitativa.

Se pudo reconocer el gesto de mano abierta o *open* en diversos ángulos. Los resultados de todos los gestos se pueden ver en la figura 4, en esta figura se observa que el sistema de envolvente convexa y la detección de sus defectos de convexidad funcionó correctamente para la detección de la mano, y como se puede ver en los casos de *open*, *point*, *grasp*, *v-pose*, *open* y *pinch* se pueden ubicar dedos correctamente. Los casos *fist*, *contain* y *pinch* fueron mas irregulares y complicados de detectar por la proximidad de los máximos de distancia entre el centro de la palma y los candidatos a dedos.

Cuadro 2. Indicaciones de disparidad para ciertas distancias

Distancia [mm]	Disparidad	Incertidumbre de la disparidad
400	99	1.2
600	-91.31	0.84

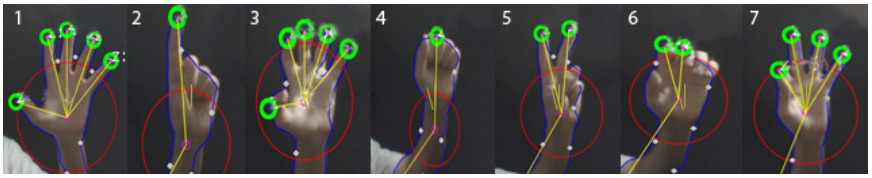

Figura 4. Gestos básicos: 1 (open), 2 (point), 3 (grasp), 4 (fist), 5 (v-pose), 6 (contain) y 7 (pinch)

La figura 5 muestra el resultado completo del proceso, donde se observa la detección de los dedos de la mano con su distancia desde el eje donde están ubicadas las cámaras expresada en milímetros. A su lado se puede ver la imagen donde se segmenta la mano por filtro HSV, vale destacar que se detectó también la cinta métrica, pero ya que esta no cumple las condiciones de la envolvente convexa no fue reconocida como "mano". Finalmente, se puede ver el mapa de disparidad, la distancia de 600mm probó estar cerca del limite máximo para el que se realizó la calibración.

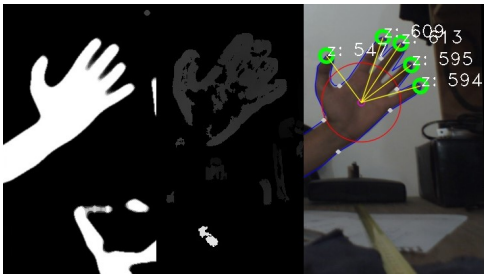

Figura 5. Proceso completo: HSV(izquierda), disparidad(centro) e imagen final(derecha)

Analizando los algoritmos utilizados, el sistema fue probado en condiciones que pueden ser desfavorables. Considerando la detección de manos se probaron distintos ángulos, la aparición de otras partes del cuerpo u objetos que tienen un color muy similar, y condiciones de iluminación desfavorables. Por el lado

del armado del mapa de disparidad, se colocaron múltiples objetos y objetos inclinados para observar gradientes sobre los mismos.

En todos los casos, el funcionamiento del sistema fue satisfactorio probando ser robusto en la mayoría estas condiciones, en la figura 6 los resultados para los diversos casos. Se observó errores en la indicación en casos donde el contorno de algún dedo se encuentra próximo al borde de un objeto del fondo. Esto se debe a que la detección de dedos se hace sobre el contorno, y en las intersecciones de bordes de objetos el mapa de disparidad no presenta el contraste suficiente.

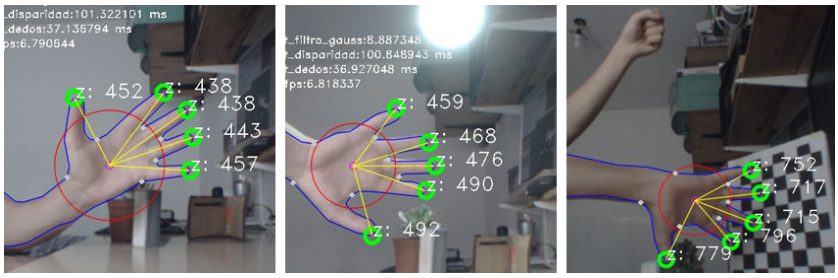

Figura 6. Condiciones desfavorables: objetos de tonalidad similar (izquierda), iluminación saturada (centro) y 2 manos con fondo complejo (derecha)

Implementado sobre un procesador Intel i5 6600K, el sistema fue capaz de procesar 7 cuadros por segundo. Se analizó el tiempo invertido por sección del procesamiento y se observó que el armado del mapa de disparidad era el lugar donde el proceso mas se demora tardando aproximadamente 100 ms por imagen, luego la detección de los dedos demora 37 ms y el resto del tiempo se reparte en los filtrados. Vale acotar que no se realizaron optimizaciones en la compilación no hace uso de multithreading ni instrucciones particulares de la arquitectura como AVX, SSE, de utilizarse se podria mejorar el valor medido.

5. Conclusiones

El sistema propuesto fue capaz de detectar la palma de la mano y de los dedos en forma tridimensional a una cadencia de 7 cuadros por segundo, siendo una buena base para aplicaciones en tiempo real.

El método de detección demostró ser inmune al ángulo en que se ubique la mano y se logró mejorar la calidad de los mapas de disparidad realizando una precalibración para obtener un set de matrices intrínsecas de las cámaras, esto además probó aumentar la repetibilidad del sistema al no requerir calibraciones en cada uso. La detección de profundidad es suficiente para realizar un control con gestos o movimientos finos (del orden de algunos mm). Para mejorar la precisión del sistema se puede reducir la distancia entre lentes y trabajar en un rango de disparidad mas adecuado para el control con manos.

El sistema fue robusto e inmune a las perturbaciones a los que se lo sometió, como otros objetos con colores similares y otras partes del cuerpo; tampoco se vió afectado por el fondo, el mismo no esta limitado a ser de un color plano y puede haber diversos objetos en el mismo como es de esperarse en el uso casual; e incluso reconoció correctamente en condiciones de iluminación desfavorables. Para reducir aún mas esto, se puede iluminar con luz infrarroja y modificar los filtros de las cámaras.

El trabajo realizado muestra la capacidad para reconocer las manos en diferentes posiciones y formas, dejando una base sólida para el reconocimiento de gestos como trabajo futuro.

Referencias

1. Jacko, J.A, y Sears, A., *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications*, 1st ed, CRC Press, 2003: 147-149.
2. Web de producto Myo. Web: https://www.myo.com/
3. Shuai Jin, Yi Li1 Guang-ming Lu, Jian-xun Luo, Wei-dong Chen, Xiao-xiang Zheng, *SOM-based Hand Gesture Recognition for Virtual Interactions*, Zhejiang University
4. John MacCormick, *How does the Kinect work?* Web: http://users.dickinson.edu/ jmac/selected-talks/kinect.pdf
5. A. Colgan, *How does the leap motion controller work*, http://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/
6. Siddharth S. Rautaray y Anupam Agrawal *Interaction with Virtual Game through Hand Gesture Recognition*, Institute of Information Technology
7. Seong-Pal Kang, Michal Tordon y Jayantha Katupitiya *Curvature Based Hand Shape Recognition for a Virtual Wheelchair Control Interface*, ARC Centre of Excellence in Autonomous Systems (CAS), School of Mechanical and Manufacturing Engineering, University of New South Wales
8. Wong Tai Man, Dr Sun Han Qiu y Dr Wong Kin Hong *ThumbStick: A Novel Virtual Hand Gesture Interface*, The Chinese University of Hong
9. Trucco y Verri, *Introductory Techniques for 3-D Computer Vision*, . Englewood Cliffs, NJ: Prentice-Hall, 1998: 178-196.
10. Documentación de la librería OpenCV. Web: http://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html
11. Aroh Barjatya, *Block Matching Algorithms For Motion Estimation*, DIP 6620 Spring 2004 Final Project Paper.
12. G. Bradsky y A. Kaehler, *Learning OpenCV*, 1st ed. O'Reilly.
13. Kohtaro Ohba, Yoichi Sato & Katsusi Ikeuchi, *Appearance-based visual learning and object recognition with illumination invariance*, Machine Vision and Applications, Springer-Verlag, 2000: 189–196
14. Chingyu Weng, Chuanyu Tseng, Mengfen Ho y Chunglin Huang *A Vision-Based Hand Motion Parameter Capturing for HCI*
15. Imran H., Anjan K. & Kandarpa K., *Hand Gesture Recognition System with Real-TimePalm Tracking*
16. Joyeeta Singha & Karen Das., *Hand Gesture Recognition Based on Karhunen-Loeve Transform*